A Formal Knowledge Level Process Model of Requirements Engineering.

Herlea, D.E.; Jonker, C.M.; Treur, J.; Wijngaards, N.J.E.

published in
Lecture Notes in Computer Science
1999

document version
Publisher's PDF, also known as Version of record

A Formal Knowledge Level Process Model of Requirements Engineering

Daniela E. Herlea[1], Catholijn M. Jonker[2], Jan Treur[2], Niek J.E. Wijngaards[1,2]

[1] Software Engineering Research Network, University of Calgary,
2500 University Drive NW, Calgary, Alberta T2N 1N4, Canada
Email: danah@cpsc.ucalgary.ca

[2] Department of Artificial Intelligence, Vrije Universiteit Amsterdam,
De Boelelaan 1081a, 1081 HV, Amsterdam, The Netherlands
URL:http://www.cs.vu.nl/~{jonker,treur, niek}. Email: {jonker,treur, niek}@cs.vu.nl

Abstract. In current literature few detailed process models for Requirements Engineering are presented: usually high-level activities are distinguished, without a more precise specification of each activity. In this paper the process of Requirements Engineering has been analyzed using knowledge-level modelling techniques, resulting in a well-specified compositional process model for the Requirements Engineering task.

1 Introduction

Requirements Engineering (RE) addresses the development and validation of methods for eliciting, representing, analyzing, and confirming system requirements and with methods for transforming requirements into specifications for design and implementation. A requirements engineering process is characterised as a structured set of activities that are followed to create and maintain a systems requirements document [4], [8], [9], [10] . To obtain insight in this process, a description of the activities is needed, the inputs and outputs to/from each activity are to described, and tools are needed to support the requirements engineering process.

No standard and generally agreed requirements engineering process exists. In [8], [10] the following activities are expected to be core activities in the process:

- *Requirements elicitation*, through which the requirements are discovered by consulting the stakeholders of the system to be developed.
- *Requirements analysis* and negotiation, through which requirements are analyzed in detail for conflict, ambiguities and inconsistencies. The stakeholders agree on a set of system requirements as well.
- *Requirements validation*, through which the requirements are checked for consistency and completeness.
- *Requirements documentation*, through which the requirements are maintenained.

In [5] also the activity *modelling* is distinguished. In [9] the main activities *elicitation*, *specification* and *validation* are distinguished. Other approaches in the literature distinguish other activities, for example, *requirements determination* [12]. These activities overlap with some of the activities mentioned above.

Various knowledge modelling methods and tools have been developed [3] and applied to complex tasks and domains. The application of a knowledge modelling method to the domain of Requirements Engineering in this paper has resulted in a compositional process model of the task of Requirements Engineering. In the literature, software environments supporting Requirements Engineering are described, but no knowledge level model is specified in detail.

requirements engineering

- 1 elicitation
  - 1.1 problem analysis
  - 1.2 elicitation of requirements and scenarios
  - 1.3 acquisition of domain ontology and knowledge
- 2 manipulation of requirements and scenarios
  - 2.1 manipulation of requirements
    - 2.1.1 detection of ambiguous and non-fully supported requirements
    - 2.1.2 detection of inconsistent requirements
    - 2.1.3 reformulation of requirements
      - 2.1.3.1 reformulation into informal requirements
      - 2.1.3.2 reformulation into semi-formal requirements
      - 2.1.3.3 reformulation into formal requirements.
    - 2.1.4 validation of requirements
    - 2.1.5 identification of clusters of requirements
  - 2.2 manipulation of scenarios
    - 2.2.1 detection of ambiguous and non-fully supported scenarios
    - 2.2.2 detection of inconsistent scenarios
    - 2.2.3 reformulation of scenarios
      - 2.2.3.1 reformulation into informal scenarios
      - 2.2.3.2 reformulation into semi-formal scenarios
      - 2.2.3.3 reformulation into formal scenarios
    - 2.2.4 validation of scenarios
    - 2.2.5 identification of clusters of scenarios
  - 2.3 identification of relationships between requirements and scenarios
- 3 maintenance of requirements and scenarios specification
  - 3.1 maintenance of requirements and scenarios documents
  - 3.2 maintenance of traceability links

Fig. 1. Overview of the process abstraction levels

In the approach presented in this paper requirements and scenarios are considered equally important. Requirements describe (e.g., functional and behavioural) properties of the system to be built, while scenarios describe use-cases of interactions between a user and the system; e.g., [6], [11]. Both requirements and scenarios can be expressed in varying degrees of formality: from informal, to semi-formal (structured natural language description), to formal (using temporal logic).

The compositional knowledge modelling method DESIRE (see [2] for the underlying principles, and [1] for a detailed case description) has been applied to obtain the formal process model the task of Requirements Engineering. The compositional

process model constructed for the Requirements Engineering task is described in Sections 2 to 5. A discussion is presented in Section 6.

2 Process Composition within Requirements Engineering

An overview of the different processes and their abstraction levels within the process Requirements Engineering is shown in Fig. 1. For each of the processes a composition relation has been specified which defines how it is composed of the processes at the next lower level of process abstraction. Note that this specification only specifies the process abstraction relations between the processes and neither the manner in which the processes interact, nor the order in which they are performed. The latter aspects are part of the process composition specifications which are discussed in Sections 2.1 and further. A specification of a process composition relation consists of a specification of the information links (static perspective) and a task control specification (dynamic perspective) which specifies when which processes and information links are performing their jobs (see [2]).

2.1 Process Composition of Requirements Engineering

The process composition of requirements engineering is described following the different levels of process abstraction depicted in Fig. 1. The composition relation (static perspective) for the first two levels of process abstraction is shown in Fig. 2: the process requirements engineering is composed of the processes elicitation, manipulation of requirements and scenarios, and maintenance of requirements and scenarios specification.

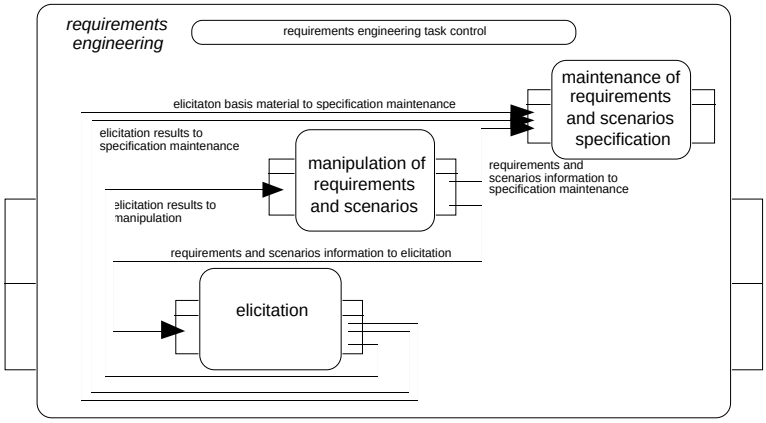

Fig. 2. Process composition of requirements engineering: information links

Within the component requirements engineering a number of information links are distinguished. The names of these information links reflect which information can be exchanged through the information link between the two processes.

The process elicitation provides initial problem descriptions, requirements and scenarios elicited from stakeholders, as well as domain ontologies and knowledge acquired in the domain. The process manipulation of requirements and scenarios manipulates

requirements and scenarios to resolve ambiguous requirements and scenarios, non-supported (by stakeholders) requirements, and inconsistent requirements and scenarios. This process reformulates requirements from informal requirements and scenarios, to more structured semi-formal requirements and scenarios, and (possibly) finally to formal requirements and scenarios. It also provides relationships among and between requirements and scenarios. The process maintenance of requirements and scenarios specification maintains the documents in which the information requirements and scenarios are described, including information on traceability.

Each of the processes depicted in Fig. 2 can be characterized in terms of their interfaces (input and output information types), as shown in Table 1.

process	*input information type*	*output information type*
elicitation	• requirements and scenarios information	• elicitation results • elicitation basic material
manipulation of requirements and scenarios	• elicitation results	• requirements and scenarios information
maintenance of requirements and scenarios specification	• elicitation results • requirements and scenarios information • elicitation basic material	• elicitation results • requirements and scenarios information • elicitation basic material

Table 1. Interface information types of direct sub-processes of requirements engineering

The dynamic perspective on the composition relation specifies control over the sub-components and information links within the component requirements engineering, as depicted in Fig. 2. Task control within requirements engineering specifies a flexible type of control: during performance of each process it can be decided to suspend the process for a while to do other processes in the mean time, and resume the original process later.

2.2 Knowledge Composition of Requirements Engineering

The information types described in the interfaces of the component requirements engineering and its direct sub-components are briefly described in this section. All of these information types specify statements *about* requirements and/or scenarios. In turn a requirement is a statement that some behavioural property is required, expressed by the object-level information types in Fig. 3. To be able to express, for example, that a requirement is ambiguous, or that a scenario has been elicited, or that a requirement is a refinement of another requirement, requirements and scenarios expressed as statements on the object level, are terms at the meta-level.

The information types specified in the interfaces of the component requirements engineering and its direct sub-components all refer to the information type requirements meta-descriptions.

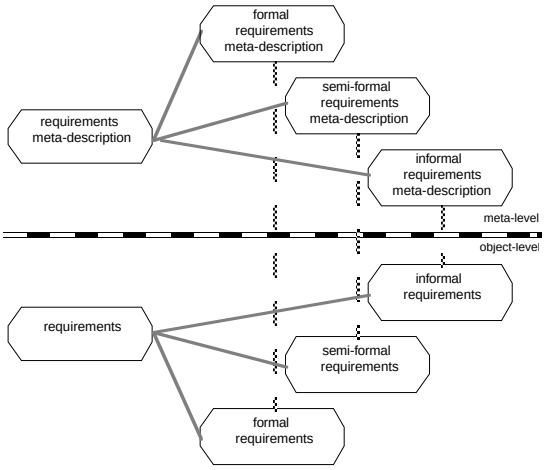

Fig. 3. Information types and meta-levels related to meta-description of requirements

The information types for scenarios are similar to the information types for requirements. The information type requirements and scenarios information is based on three information types: requirements information, scenarios information, and relations between requirements and scenarios. In turn, the information type requirements information is based on three information types: current requirements, clusters of requirements, and relations among requirements. The information type scenarios information is based on three similar information types: current scenarios, clusters of scenarios, and relations among scenarios.

3 Composition of Elicitation

The first two levels of process abstraction for elicitation are shown in Fig. 4. The processes problem analysis, acquisition of domain ontology and knowledge, and elicitation of requirements and scenarios are distinguished within the process elicitation.

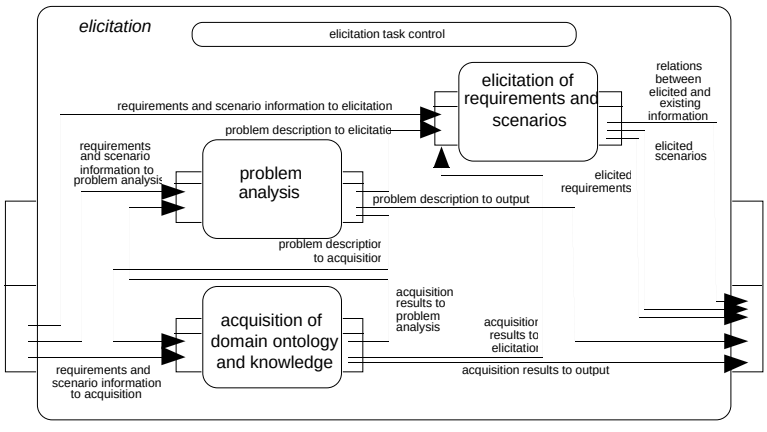

Fig. 4. Process composition relation of elicitation : information links

The three sub-processes of elicitation, as depicted in Fig. 4, are closely intertwined. The process problem analysis extracts the (initial) perceived problem from the stakeholders. It can also determine that requirements and scenarios are needed for another level of process abstraction. The process acquisition of domain ontology and knowledge acquires from stakeholders ontologies and knowledge of the domain, possibly related to existing requirements and scenarios. The process elicitation of requirements and scenarios elicits requirements and scenarios from stakeholders on the basis of identified problems, existing requirements and scenarios. Each of the processes depicted in Fig. 4 can be characterized in terms of their interface information types, as shown in Table 2.

process	*input information type*	*output information type*
acquisition of domain ontology and knowledge	• requirements and scenarios information • problem description	• acquisition results
problem analysis	• requirements and scenarios information • acquisition results	• problem description
elicitation of requirements and scenarios	• requirements and scenarios information • acquisition results • problem description	• elicited requirements • elicited scenarios • relations between elicited and existing information

Table 2. Input and output information types of the direct sub-processes of the process elicitation.

The dynamic perspective on the process composition within elicitation, task control, specifies flexible control, similar to the control one process abstraction higher.

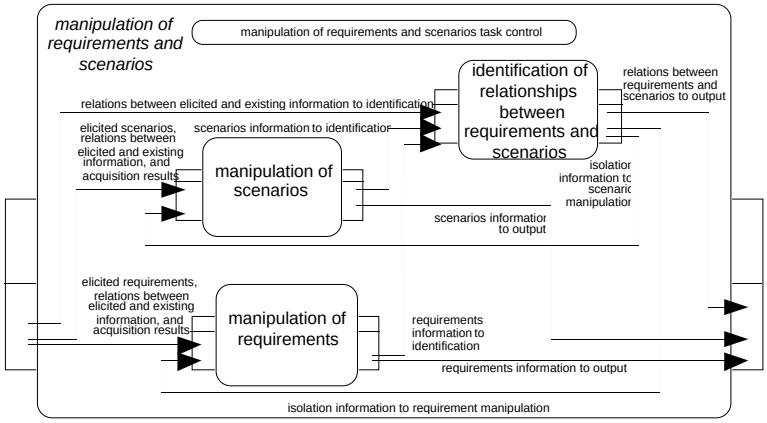

Fig. 5. Process composition of manipulation of requirements and scenarios: information links.

4 Composition of Manipulation of Requirements and Scenarios

The process composition relation between the first two levels of process abstraction for manipulation of requirements and scenarios are shown in Fig. 5. The processes manipulation of requirements, manipulation of scenarios, and identification of relationships between requirements and scenarios are distinguished within the process manipulation of requirements and scenarios.

The process manipulation of requirements is responsible for removing ambiguities, resolving non-fully supported requirements (by stakeholders), and resolving inconsistencies, while striving for progressive formalisation of requirements. This process also produces the relationships among requirements. The process manipulation of scenarios is similar to the process manipulation of requirements. The process identification of relationships between requirements and scenarios establishes which requirements are related to which scenarios, and vice versa. Each of the processes depicted in Fig. 5 can be characterized in terms of their interface information types, as shown in Table 3.

process	*input information type*	*output information type*
manipulation of requirements	• elicited requirements • relations between elicited and existing information • acquisition results • isolation information	• requirements information
manipulation of scenarios	• elicited scenarios • relations between elicited and existing information • acquisition results • isolation information	• scenarios information
identification of relationships between requirements and scenarios	• requirements information • scenarios information	• relations between requirements and scenarios • isolation information

Table 3. Interface information types of the processes within manipulation of requirements and scenarios.

Also in this case the dynamic perspective on the composition relation specifies flexible control over the sub-components of the component manipulation of requirements and scenarios.

5 Composition of Manipulation of Requirements

The first level of process abstraction within manipulation of requirements is shown in Fig. 6. The processes reformulation of requirements, validation of requirements, detection of ambiguous and non-fully supported requirements, detection of inconsistent requirements, and identification of functional clusters of requirements are distinguished within the process manipulation of requirements.

The process detection of ambiguous and non-fully supported requirements analyses the requirements for ambiguities and the extent of non-supportedness of requirements (by stakeholders). The process detection of inconsistent requirements analyses the requirements

for inconsistencies among requirements. The process reformulation of requirements plays an important role within manipulation of requirements: problematic requirements are reformulated into (less problematic) requirements by adding more and more structure to requirements: from informal to semi-formal to formal. The process validation of requirements has interaction with stakeholders to establish the supportedness of a requirement in relation to a stakeholder, and whether pro and con arguments exist for a requirement. The process identification of clusters of requirements identifies clusters of requirements on the basis of clustering criteria.

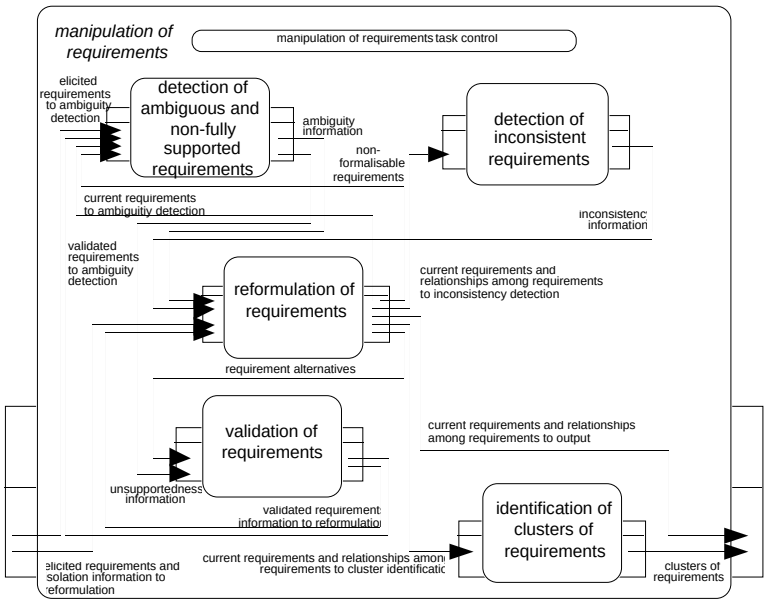

Fig. 6. Process composition of manipulation of requirements: information links.

As before, each of the processes depicted in Fig. 6 can be characterized in terms of their interface information types, as shown in Table 4. Also in this case, task control specifies flexible control. The process manipulation of scenarios has a structure similar to manipulation of requirements.

6 Discussion

The compositional knowledge modelling method DESIRE has been applied to the task of Requirements Engineering. The resulting compositional process model has been presented in some detail in this paper. The process model has been constructed on the basis of studies of available literature, and the application of requirements engineering techniques to a real-life case study: the design of a Personal Internet Assistant [7].

process	*input information type*	*output information type*
detection of ambiguous and non-fully supported requirements	• elicited requirements • current requirements • validated requirements information • non-formalisable requirements	• ambiguity information • unsupportedness information
detection of inconsistent requirements	• current requirements • relations among requirements	• inconsistency information
reformulation of requirements	• elicited requirements • ambiguity information • inconsistency information • validated requirements information • isolated scenarios	• current requirements • relations among requirements • requirement alternatives • non-formalisable requirements
validation of requirements	• requirement alternatives • unsupportedness information	• validated requirements information
identification of clusters of requirements	• current requirements • relations among requirements	• clusters of requirements

Table 4. Interface information types of processes within manipulation of requirements.

The processes have been described at different levels of process abstraction, with descriptions of their interfaces, a static composition relation (possibilities for information exchange) and a dynamic composition relation ('control flow'). The static composition relation does not prescribe a particular task control through the process composition. The task control is formulated in terms of conditions which trigger particular activities. The task control specification reflects the amount of flexibility and iterative nature of sub-processes of the requirements engineering process.

The compositional process model presented in this paper has been formally specified and provides more details and structure for the requirements engineering process than process models described in the literature on requirements engineering. For example, in [8], [10] the following activities are considered core activities in the requirements engineering process: 'requirements elicitation', 'requirements analysis and negotiation', 'requirements documentation', and 'requirements validation'. The first three of these core activities form the top level composition of the process model introduced in this paper. In contrast to the references mentioned, in the model introduced here a detailed specialisation of these three main processes is added. In the process model introduced the fourth main activity, 'requirements validation' is considered an integrated part of the manipulation processes both for requirements and scenarios, and is modelled within these processes: detection of inconsistent requirements, detection of inconsistent scenarios, validation of requirements, validation of scenarios.

To further investigate the applicability of this compositional process model, additional requirements engineering experiments will be conducted. The formally specified compositional process model for the task of requirements engineering can be employed in the design of automated tools for requirements engineering (e.g., [5]),

supporting the activities of (human) requirement engineers on the basis of an agreed shared model of the requirements engineering task.

References

1. Brazier F M T, Dunin-Keplicz B M, Jennings N R, and Treur J., Formal Specification of Multi-Agent Systems: a Real World Case In: Lesser V (ed.) *Proceedings First International Conference on Multi-Agent Systems ICMAS'95* (1995) pp. 25-32 MIT Press. Extended version in: Huhns M and Singh M (eds.) *International Journal of Co-operative Information Systems IJCIS* Vol. 6 No 1 (1997) pp. 67-94.
2. Brazier, F.M.T., Jonker, C.M., and Treur, J., Principles of Compositional Multi-agent System Development. In: J. Cuena (ed.), *Proceedings of the 15th IFIP World Computer Congress, WCC'98, Conference on Information Technology and Knowledge Systems, IT&KNOWS'98* , 1998, pp. 347-360.
3. Brazier, F.M.T., and Wijngaards, N.J.E., An instrument for a purpose driven comparison of modelling frameworks. In: Plaza, E., and Benjamins, R. (eds.). *Proceedings of the 10th European Workshop on Knowledge Acquisition, Modelling, and Management (EKAW'97)*. Sant Feliu de Guixols, Catalania, Lecture Notes in Artificial Intelligence, vol. 1319, Springer Verlag, 1997, pp. 323-328.
4. Davis, A. M., Software requirements: Objects, Functions, and States, Prentice Hall, New Jersey, 1993.
5. Dubois, E., Du Bois, P., and Zeippen, J.M., A Formal Requirements Engineering Method for Real-Time, Concvurrent, and Distributed Systems. In: *Proc. of the Real-Time Systems Conference*, RTS'95, 1995.
6. Erdmann, M. and Studer, R., Use-Cases and Scenarios for Developing Knowledge-based Systems. In: *Proc. of the 15th IFIP World Computer Congress, WCC'98, Conference on Information Technologies and Knowledge Systems, IT&KNOWS'98* (J. Cuena, ed.), 1998, pp. 259-272.
7. Herlea, D., Jonker, C.M., Treur, J. and Wijngaards, N.J.E., *A Case Study in Requirements Engineering: a Personal Internet Agent*. Technical Report, Vrije Universiteit Amsterdam, Department of Artificial Intelligence, 1999. URL: http://www.cs.vu.nl/~treur/pareqdoc.html
8. Kontonya, G., and Sommerville, I., *Requirements Engineering: Processes and Techniques*. John Wiley and Sons, New York, 1998.
9. Loucipoulos, P. and Karakostas, V., *System Requirements Engineering*. McGraw-Hill, London, 1995.
10. Sommerville, I., and Sawyer P., *Requirements Engineering: a good practice guide*. John Wiley & Sons, Chicester, England, 1997.
11. Weidenhaupt, K., Pohl, M., Jarke, M. and Haumer, P., Scenarios in system development: current practice, in *IEEE Software*, pp. 34-45, March/April, 1998.
12. Yadav, S. et al., Comparison of analysis techniques for information requirements determination. Communications of the ACM, vol. 27 (2), 1988.